Evolutionary Computing as a Tool for Grammar Development

Guy De Pauw

CNTS – Language Technology Group
UIA – University of Antwerp
Antwerp – Belgium
guy.depauw@ua.ac.be

Abstract. In this paper, an agent-based evolutionary computing technique is introduced, that is geared towards the automatic induction and optimization of grammars for natural language (GRAEL). We outline three instantiations of the GRAEL-environment: the GRAEL-1 system uses large annotated corpora to bootstrap grammatical structure in a society of autonomous agents, that tries to optimally redistribute grammatical information to reflect accurate probabilistic values for the task of parsing. In GRAEL-2, agents are allowed to mutate grammatical information, effectively implementing grammar rule discovery in a practical context. Finally, by employing a separate grammar induction module at the onset of the society, GRAEL-3 can be used as an unsupervised grammar induction technique.

1 Introduction

An important trend in the field of Machine Learning sees researchers employing combinatory methods to improve the classification accuracies of their algorithms. Natural language problems in particular benefit from combining classifiers to deal with the large datasets and expansive arrays of features that are paramount in describing this difficult and disparate domain that typically features a considerable amount of sub-regularities and exceptions [1]. Not only system combination and cascaded classifiers are well-established methods in the field of Machine Learning for natural language [2,3], also the techniques of bagging and boosting [4] have been used successfully on a number of natural language classification tasks [5,6]. These techniques hold in common that in no way do they alter the actual content of the information source of the predictor. Simply by re-distributing the data, different resamplings of the same classifier are generated to create a combination of classifiers.

The field of evolutionary computing has been applying problem-solving techniques that are similar in intent to the aforementioned Machine Learning recombination methods. Most evolutionary computing approaches hold in common that they try and find a solution to a particular problem, by recombining and mutating individuals in a society of possible solutions. This provides an attractive technique for problems involving large, complicated and non-linearly divisible search spaces. The evolutionary computing paradigm has however always seemed reluctant to deal with issues of natural language syntax. The fact that syntax is in essence a recursive, non-propositional system, dealing with complex issues such as long-distance dependencies and constraints, has made it difficult to incorporate it in typically propositional evolutionary systems such as genetic algorithms. Most GA syntactic research so far has focused on non-linguistic data, with some notable exceptions [7,8,9,10]. Yet none of these systems are suited to a generic grammar optimization task, mainly because the grammatical formalism and evolutionary processes underlying these systems are designed to fit a particular task, such as information retrieval [11]. Yet so far, little or no progress has been achieved in evaluating evolutionary computing as a tool for the induction or optimization of data-driven parsing techniques.

The GRAEL (**GRA**mmar **E**vo**L**ution) framework [12] attempts to combine the sensibilities of the recombination machine learning methods and the attractive evolutionary properties of the concepts of genetic programming. It provides a suitable framework for the induction and optimization of any type of grammar for natural language in an evolutionary setting. In this paper we want to provide a general overview of GRAEL as a natural language grammar development technique. We will first identify the basic problem in Section 2, after which we outline the general architecture of the GRAEL environment in Section 3. Next, we will introduce three different instantiations of the GRAEL environment: in GRAEL-1 (Section 4) large annotated corpora are used to bootstrap grammatical structure in a society of agents, who engage in a series of communicative attempts, during which they redistribute grammatical information to reflect optimized probabilistic values for the task of parsing. In GRAEL-2 (Section 5), agents are allowed to mutate grammatical information, effectively implementing grammar rule discovery in a practical context. Finally, we look at GRAEL-3 in Section 6, which provides a method for unsupervised grammar induction.

2 Natural Language Grammar Development

Syntactic processing has always been deemed to be paramount to a wide range of applications, such as machine translation, information retrieval, speech recognition and the like. It is therefore not surprising that natural language syntax has always been one of the most active research areas in the field of language technology. All of the typical pitfalls in language like ambiguity, recursion and long-distance dependencies, are prominent problems in describing syntax in a computational context. Historically, most computational systems for syntactic *parsing*, employ hand-written grammars, consisting of a laboriously crafted set of grammar rules to apply syntactic structure to a sentence[1]. But in recent years, a lot of research efforts are trying to automatically induce workable grammars from annotated corpora, i.e. large collections of pre-parsed sentences [13]. Since the tree-structures in these annotated corpus already implicitly contain a grammar, it is a relatively trivial task to induce a large-scale grammar and parser that is able to acquire reasonably high parsing accuracies on a held-out set of data [14,15,16].

Yet, data-analysis of the output generated by these parsers still brings to light fundamental limitations to these corpus-based methods. Even though they generally provide a much broader coverage as well as higher accuracy than hand-built grammars, corpus-induced grammars will still not hold enough grammatical information to provide

[1] Syntactic structure is typically presented as a parse tree, such as the ones in Figure 1.

structures for a large number of sentences in language, as some rules that are needed to generate the correct tree-structures are not induced from the original corpus. But even if there were such a thing as a full-coverage corpus-induced grammar, performance would still be limited by the probabilistic weights attributed to its rules. The GRAEL system described in this paper tries to alleviate the problems inherent to corpus-induced grammars, by establishing a distributed evolutionary computing method for grammar induction and optimization. Generally, GRAEL can be considered as a system that allows for the simultaneous development of a range of alternative solutions to a grammatical problem, optimized in a series of practical interactions in a society of agents controlled by evolutionary parameters.

3 Grammar Evolution

A typical GRAEL society consists of a population of agents in a virtual environment, each of which holds a number of structures that allow them to generate sentences as well as analyze other agents' sentences. These grammars are updated through an extended series of inter-agent interactions, using a form of error-driven learning. The evolutionary parameters are able to define the content and quality of the grammars that are being developed over time, by imposing fitness functions on the society. By embedding the grammars in autonomous agents, GRAEL ensures that the grammar development is grounded in the practical task of parsing itself.

The grammatical knowledge of the agents is typically bootstrapped by using an annotated natural language corpus [13]. At the onset of such a corpus-based GRAEL society, the syntactic structures of the corpus are randomly distributed over the agents, so that each agent holds a number of tree-structures in memory. The actual communication between agents is implemented in *language games* [17]: an agent (**ag1**) presents a sentence to another agent (**ag2**). If **ag2** is able to correctly analyze **ag1**'s sentence, the communication is successful. If on the other hand, **ag2** is lacking the proper grammatical information to parse the sentence correctly, **ag1** shares the necessary information for **ag2** to arrive at the proper solution.

A Toy Example

We take a look at an example of a very basic language game: Figure 1 shows a typical interaction between two agents. In this example, an annotated corpus of two sentences has been distributed over two agents. The two agents engage in a language game, in which **ag1** provides an assignment to **ag2**: **ag1** presents the sentence "I offered some bear hugs" to **ag2** for parsing. **ag2**'s knowledge does not contain the proper grammatical information to interpret this sentence the way **ag1** intended and so **ag2** will return an incorrect parse, albeit consistent with its own grammar.

ag1 will consequently try and help **ag2** out by revealing the minimal correct substructure of the correct parse that should enable **ag2** to arrive at the correct solution. **ag2** will incorporate this information in its grammar and try to parse the sentence again with the updated knowledge. Once **ag2** is able to provide the correct analysis (or is not able to after a certain number of attempts) either **ag1**'s next sentence will be parsed, or two other agents in the GRAEL society will be randomly selected to play a language game.

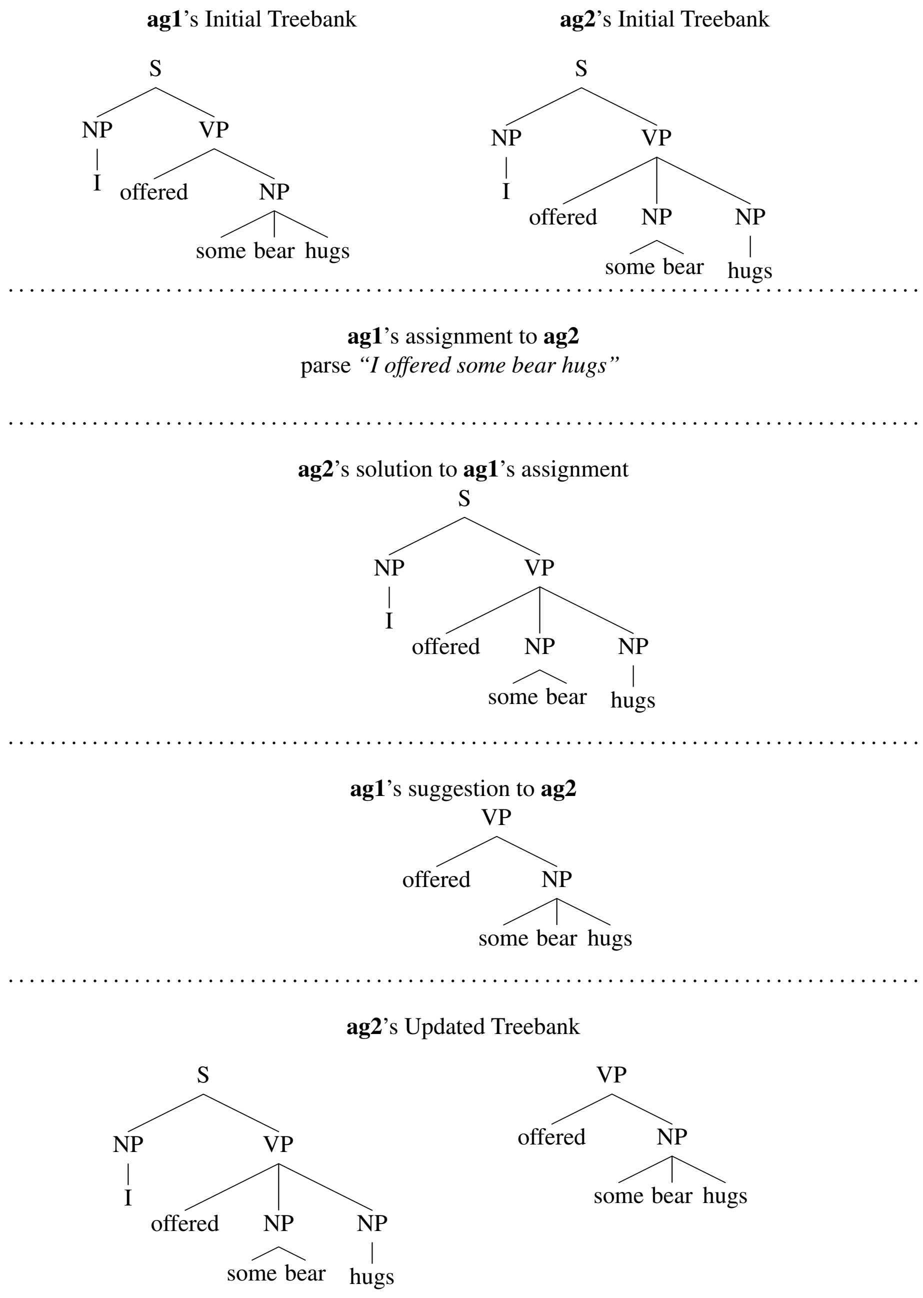
ag1's Initial Treebank
ag2's Initial Treebank
S
NP
VP
I
offered
NP
some bear hugs
ag1's assignment to **ag2**
parse *"I offered some bear hugs"*
ag2's solution to **ag1**'s assignment
ag1's suggestion to **ag2**
ag2's Updated Treebank

Fig. 1. A GRAEL language game

Generations

This type of interaction is based on a high amount of knowledge sharing between agents. This extends the agents' grammars very fast, so that their datasets can grow very large in a short period of time. It is therefore beneficial to introduce new generations in the GRAEL society from time to time. This not only allows for tractable computational processing times, but also allows the society to purge itself of bad agents and build new generations of good parser agents, who contain a fortuitous distribution of grammatical knowledge.

This introduces a neo-darwinist aspect in the system and involves the use of fitness functions that can distinguish good agents from bad ones. Typically, we define the fitness of an agent in terms of its parsing accuracy (i.e. the number of correct analyses), but we can also require the agents to have fast and efficient grammars and the like. The use of fitness functions and generations ideally makes sure that the required type of grammatical knowledge is retained throughout different generations, while useless grammatical knowledge can be marginalized over time.

Unfortunately, it is not feasible to provide a detailed description of the evolutionary parameters in the context of this overview paper. We would like to refer to [12] for specific details on the architecture and parameters of the GRAEL environment.

4 GRAEL-1: Probabilistic Grammar Optimization

GRAEL-I is the most straightforward instantiation of GRAEL and deals with probabilistic grammar optimization. Developing a corpus-based parser requires inducing a grammar from an annotated corpus and using it to parse new sentences. Typically, these grammars are very large, so that for any given sentence a huge amount of possible parses is generated (a *parse forest*), only one of which provides the correct analysis for the sentence. Parsing can therefore be considered as a two-step process: first a parser generates all possible parses for a sentence, after which a disambiguation step makes sure the correct analysis is retrieved from the parse forest. Fortunately, we can also induce probabilistic information from the annotated corpus[2] that provides a way to rank the analyses in the parse forests in order of probabilistic preference. Even though these statistics go a long way in providing well ordered parse forests, it can be observed in many cases that the ranking of the parse forest is sometimes counter-intuitive in that correct constructs are often overtaken by obviously erroneous, but highly frequent structures.

With GRAEL-I, we propose an agent-based evolutionary computing method to resolve the issue of suboptimal probability mass distribution: by distributing the knowledge over a group of agents and having them interact with each other, we basically create a multiple-route model for probabilistic grammar optimization. Grammatical structures extracted from the training corpus, will be present in different quantities and variations throughout the GRAEL society (similarly to the aforementioned machine learning method of *bagging*). While the agents interact with each other and in effect practice the task on each other's grammar, a varied range of probabilistic grammars are optimized in a situation that directly relates to the task at hand (similarly to the machine learning method of *boosting*).

[2] This is achieved by observing the relative frequency of grammatical constructs in the annotated corpus.

4.1 Experimental Setup

In the GRAEL-I experiments, we measure two types of accuracy: the baseline accuracy is measured by directly inducing a grammar from the training set to power a parser, which disambiguates the test set. The same training set is then randomly distributed over a number of agents in the GRAEL society, who will consequently engage in a number of *language games*. At some point, the society is halted and the fittest agent is selected from the society. This agent effectively constitutes a redistributed and probabilistically optimized grammar, which can be used to power another parser. GRAEL-I accuracy is achieved by having this parser disambiguate the same test set as the baseline parser.

Data and tools. Two data sets from the Penn Treebank [13] were used. The main batch of experiments was conducted on an edited version of the small, homogeneous ATIS-corpus, which consists of a collection of annotated sentences recorded by a spoken-dialogue system. The larger Wall Street Journal Corpus (henceforth WSJ), a collection of annotated newspaper articles, was used to test the system on a larger scale corpus. We used the parsing system PMPG [16], which combines a CKY parser [18] and a post-parsing parse forest reranking scheme that employs probabilistic information as well as a memory-based operator that ensures that larger syntactic contexts are considered during parsing.

Apart from different corpora, we also experimented on different society sizes (5, 10, 20, 50, 100 agents), generation methods, fitness functions and methods to determine when to halt a GRAEL society. An exhaustive overview of all experimental parameters can be found in [12], but we will briefly outline some key notions. New generations are created as follows: if an agent is observed not to acquire any more rules over the course of *n* communicative attempts[3], it is considered to be an end-of-life agent. As soon as two end-of-life agents are available that belong to the 50% fittest agents in the society, they are allowed to procreate by crossing over the grammars they have acquired during their lifespan. This operation yields three new agents, two of which will take their ancestors' slots in the society, while the other one takes the slot of the oldest agent among the 50% unfit agents at that point in time. The fitness of an agent is defined by recording a weighted average of the F-score (see below) during inter-agent communication and the F-score of the agent's parser on a held-out validation set. This information was also used to try and halt the society at a global maximum and to select the fittest agent from the society. For computational reasons, the experiments on the WSJ-corpus were limited to two different population sizes (50 and 100) and used an approximation of GRAEL that can deal with large datasets in a reasonable amount of time.

Note that GRAEL-I includes two different notions of crossover, although either one is a far stretch from the classic GA-type definition of the concept. The first type of crossover occurs during the language game when parts of syntactic tree-structures are being shared between agents. This operation relates to the recombination of knowledge aspect of crossover. The second type of crossover occurs when new agents are created by crossing over the grammars of two end-of-life agents. Again, the aspect of recombination is apparent in this operation. Note however the distinction with crossover in the context

[3] *n* is by default set to the number of agents in a society.

of genetic algorithms in that neither crossover operation in the GRAEL system occurs on the level of the genotype and that it is in fact the phenotype that is being adapted and which evolves over time[4].

Table 1. Baseline vs. GRAEL-I results

	ATIS		WSJ	
	Exact Match	$F_{\beta=1}$-score	Exact Match	$F_{\beta=1}$-score
Baseline	70.7	89.3	16.0	80.5
GRAEL (5)	72.4	90.9	—	—
GRAEL (10)	77.6	92.1	—	—
GRAEL (20)	77.6	92.1	—	—
GRAEL (50)	75.9	92.2	22.2	80.7
GRAEL (100)	75.9	92.0	22.8	81.1

4.2 Results

Table 1 displays the results of these experiments. Exact Match Accuracy expresses the percentage of sentences that were parsed completely correct, while the F-score is a measure of how well the parsers work on a constituent level. The baseline model is a standard PMPG parser using a grammar directly induced from the training set. Table 1 also displays scores of the GRAEL system for different population sizes. We notice a significant gain for all GRAEL models over the baseline model on the ATIS corpus. The small society of 5 agents achieves only a very limited improvement over the baseline method. Data analysis showed that the best moment to halt the society and select the fittest agent from the society, is a relatively brief period right before actual convergence sets and grammars throughout the society are starting to resemble each other more closely. The size of the the society seems to be the determining factor controlling the duration of this period. In smaller societies, it may occur that convergence sets in too fast, since there is a narrower spread of data throughout the society. This causes convergence to set in prematurely, before the halting procedures even had a chance to register a proper halting point for the society. Hence, the low accuracy for the 5 agent society on the ATIS corpus.

Some preliminary experiments on a subset of the WSJ corpus had shown that society sizes of 20 agents and less to be unsuitable for a large-scale corpus, again ending up in a harmful premature stagnation. The gain achieved by the GRAEL society is less spectacular than on the ATIS corpus, but it is still statistically significant. Larger society sizes and full GRAEL processing on the WSJ corpus should achieve a larger gain, but is not currently feasible due to computational constraints.

The results show that GRAEL-I is indeed an interesting method for probabilistic grammar redistribution and optimization. Data analysis shows that many of the counter-intuitive parse forest orderings that were apparent in the baseline model, are being

[4] Theoretically, this method relates to the empiricist point of view of language acquisition, rather than the nativist point of view.

resolved after GRAEL-1 processing. It is also interesting to point out that we are achieving an error reduction rate of more than 26% over the baseline method, without introducing any new grammatical information in the society, but solely by redistributing what is already there.

5 GRAEL-2: Grammar Rule Discovery

Any type of grammar, be it corpus-induced or hand-written will not be able to cover all sentences of language. Some sentences will indeed require a rule that is not available in the grammar. Even for a large corpus such as the WSJ, missing grammar rules provide a serious accuracy bottleneck. We therefore set out to find a method that can take a grammar and improve its coverage by generating new rules. But doing so in an unguided manner, would yield huge, over-generating grammars, containing many nonsensical rules. The GRAEL-2 system described in this section provides a guidance mechanism to grammar rule discovery. In GRAEL-2, the original grammar is distributed among a group of agents, who can randomly *mutate* the grammatical structures they hold. The new grammatical information they create is tried and tested by interacting with each other. The neo-darwinist aspect of this evolutionary system tries to retain any useful mutated grammatical information throughout the population, while noise is filtered out over time. This method provides a way to create new grammatical structures previously unavailable in the corpus, while at the same time evaluating them in a practical context, without the need for an external information source.

Some minor alterations need to be made to the initial GRAEL-1 system to accomplish this, most notably the addition of an element of mutation. This occurs in the context of a language game (cf. Figure 1) at the point where **ag1** suggests the minimal correct substructure to **ag2**. In GRAEL-1 this step introduced a form of error-driven learning, making sure that the probabilistic value of this grammatical structure is increased. The functionality of GRAEL-2 however is different: we assume that there is a virtual noisy channel between **ag1** and **ag2** which may cause **ag2** to misunderstand **ag1**'s structure. Small mutations on different levels of the substructure may occur, such as the deletion, addition and replacement of nodes in the tree-structure. This mutation introduces previously unseen grammatical data in the GRAEL society, some of which will be useless (and will hopefully disappear over time), some of which will actually constitute good grammar rules.

Note again that the concept of mutation in the GRAEL system, stretches the classic GA-notion. Mutation in GRAEL-2 does not occur on the level of the genotype at all. It is the actual grammatical information that is being mutated and consequently communicated throughout the society. This provides a significant speed-up of grammatical evolution over time, as well as enable a transparent insight into the grammar rule discovery mechanism itself.

Experimental Setup and Results. The GRAEL-2 experiments have a similar setup to the GRAEL-1 experiments (Section 4). For the experiments on the ATIS corpus, we compiled a special worst-case scenario test set to specifically test the grammar-rule discovery capabilities of GRAEL-2. This test set consists of 97 sentences that require a grammar

Table 2. Baseline vs GRAEL-1 vs GRAEL-2 vs GRAEL2+1 Results

	ATIS		WSJ	
	$F_{\beta=1}$	**Ex. Match**	$F_{\beta=1}$	**Ex. Match**
Baseline	69.8	0	80.5	16
grael-1	73.8	0	81.4	22.8
grael-2	83.0	7.2	76.5	19.3
grael2+1	85.7	11.3	81.6	23.4

rule that cannot be induced from the training set. For the WSJ-experiments the standard test set was used. A 20-agent and a 100 agent society were respectively used for the ATIS and WSJ experiments.

Table 2 compares the GRAEL-2 results to the baseline and GRAEL-1 systems. The latter systems trivially achieve an exact match accuracy of 0% on the ATIS test set, which also has a negative effect on the F-score (Table 2). GRAEL-2 is indeed able to improve on this significantly. The results on the WSJ corpus show however that GRAEL-2 has lost the beneficial probabilistic optimization effect that was paramount to GRAEL-1. Another experiment was therefore conducted in which we turned the GRAEL-2 society into a GRAEL-1 society after the former's halting point. In other words: we take a society of agents using mutated information and consequently apply GRAEL-1 probabilistic redistribution on these grammars. This achieves a significant improvement on all data sets and establishes an interesting grammar development technique that is able to extend and optimize any given grammar without the need for an external information source.

6 GRAEL-3: Unsupervised Grammar Induction

In GRAEL-2 we started off with an initial grammar that is induced from an annotated corpus, so that we could consider it to be a form of supervised grammar induction, since we still require annotated data. Yet, annotated data is hard to come by and resources are necessarily limited. Recent research efforts however try to implement methods that can apply structure to raw data, simply on the basis of distributional properties and grammatical principles [19,20,21]. Further alterations to the GRAEL-2 system can extend its functionality to include this type of task.

The GRAEL-3 system requires us to develop a basic, but workable grammar induction module, that can build tree-structures on the basis of mutual information content calculated on bigrams. This module should not be considered a part of GRAEL-3 proper: any kind of grammar induction method can in principle be used to bootstrap structure in the society. Next, we performed three types of experiment for each data set: GRAEL-3A takes the whole training set and applies structure to those sentences based on information content values calculated on the entire data set. GRAEL-3B first distributes the sentences over the agents, after which the grammar induction module applies structure to these sentences based on information content values calculated for each agent individually. Since this grammar induction module effectively constitutes a parser, we also conducted some experiments in which parsing was only performed using this method (GRAEL-3AB-2).

The same training set/test set divisions were used as in the GRAEL-1 experiments, while the experiments were performed on a 20-agent society for the ATIS-corpus and a 100-agent society for the WSJ-corpus. Due to reasons of time, we did not perform the baseline experiment using the PMPG for the WSJ experiments, nor the GRAEL-3A-1 experiment. We measure the F-score and the zero-crossing brackets measure which is typically used to evaluate unsupervised grammar induction methods.

Table 3. GRAEL-3 Results

	ATIS		WSJ	
	$F_{\beta=1}$	0CB	$F_{\beta=1}$	0CB
Baseline (PMPG)	22.4	22.9	–	–
GRAEL-3A-1	25.6	24.9	–	–
GRAEL-3B-1	22.7	22.9	31.8	32.8
Baseline (GIM)	28.4	30.8	32.2	32.5
GRAEL-3AB-2	31.0	31.1	33.8	34.0

The first line of Table 3 shows the baseline accuracy using a PMPG on the training set annotated by the grammar induction module. These figures are very low, which is mainly due to the problematic labeling properties the grammar induction method imposes. A parser using ps-type rules such as PMPG does indeed need accurate node labels to be able to process grammatical structure in an accurate manner. The GRAEL-3A-1 system however is able to improve on this grammar significantly. Unsupervised grammar induction methods typically need a lot of data to achieve reasonable accuracy. It is therefore not surprising that GRAEL-3B does not achieve any improvement over the baseline, since the grammar induction method only has a very limited amount of data to extract useful information content values from. Using the grammar induction method as a parser itself circumnavigates the problem of labeling and this has a positive effect on parsing accuracy. More importantly, GRAEL-3 seems again able to improve parsing accuracy significantly, both on the ATIS and the WSJ corpus.

7 Concluding Remarks

This paper presented a broad overview of the GRAEL system, which can be used for different grammar optimization and induction tasks. We believe this to be one of the first research efforts that employs agent-based evolutionary computing as a machine learning method for data-driven grammar development. Using the same architecture and only applying minor alterations, we were able to implement three different tasks: GRAEL-1 provides a beneficial re-distribution of the probability mass of a probabilistic grammar by using a form of error-driven learning in the context of interactions between autonomous agents. By introducing an element of mutation, we extended GRAEL-1's functionality and projected GRAEL-2 as a workable grammar rule discovery method, significantly improving grammatical coverage on corpus-induced grammars. Following up GRAEL-2's grammar rule discovery method with GRAEL-1's probabilistic grammar optimization proved to be an interesting optimization toolkit for corpus-induced grammars. Finally, we described GRAEL-3 as a first attempt to provide an unsupervised grammar induction technique. Even though the scores achieved by GRAEL-3 are rather modest compared to supervised approaches, the experiments show that the GRAEL environment is again able to take a collection of deficient grammars and turn them into better grammars through an extended process of inter-agent interaction.

The GRAEL framework provides an agent-based evolutionary computing approach to natural language grammar optimization and induction. It integrates the sensibilities of combinatory machine learning methods such as bagging and boosting, with the dynamics of evolutionary computing and agent-based processing. We have shown that GRAEL-1 and GRAEL-2 are able to take a collection of annotated data, providing an already well-balanced grammar, and squeeze more performance out of them without using an external information source. The experiments with GRAEL-3 showed however that is equally able to improve on a collection of poor initial grammars, proving that the GRAEL framework is indeed able to provide an optimization for any type of grammar, regardless of its initial quality. This projects GRAEL as an interesting workbench for natural language grammar development, both for supervised, as well as unsupervised grammar optimization and induction tasks.

Acknowledgments. The research described in this paper was financed by the FWO (Fund for Scientific Research). The author would like to acknowledge Frederic Chappelier, for kindly making his parser available [18].

References

1. Daelemans, W., van den Bosch, A., Zavrel, J.: Forgetting exceptions is harmful in language learning. Machine Learning, Special issue on Natural Language Learning **34** (1999) 11–41
2. van Halteren, Hans, J.Z., Daelemans, W.: Improving accuracy in word class tagging through combination of machine learning systems. Computational Linguistics **27 (2)** (2001) 199–230
3. Tjong Kim Sang, E., Daelemans, W., D'ejean, W., Koeling, H., Krymolowski, R., Punyakanok, Y., Roth, V.: Applying system combination to base noun phrase identification. In: Proceedings of COLING 2000, Saarbruecken, Germany (2000) 857–863
4. Breiman, L.: Bagging predictors. Machine Learning **24** (1996) 123–140
5. Abney, S., Schapire, R., Singer, Y.: Boosting applied to tagging and pp attachment. In: Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora. (1999) 38–45
6. Henderson, J., Brill, E.: Bagging and boosting a treebank parser. In: Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-2000). (2000) 34–41
7. Smith, T.C., Witten, I.H.: Learning language using genetic algorithms. In Wermter, S., Riloff, E., Scheler, G., eds.: Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing. Volume 1040 of LNAI. Springer Verlag, Berlin (1996) 132–145
8. Wyard, P.: Context-free grammar induction using genetic algorithms. In Belew, R., Booker, L., eds.: Proceedings of the Fourth International Conference on Genetic Algorithms, San Mateo, ICGA, Morgan Kaufmann (1991) 514–518

9. Antonisse, H.J.: A grammar-based genetic algorithm. In Rawlings, G.J.E., ed.: Foundations of genetic algorithms. Morgan Kaufmann, San Mateo (1991) 193–204
10. Araujo, L.: A parallel evolutionary algorithm for stochastic natural language parsing. In: Proceedings of The Seventh International Conference on Parallel Problem Solving From Nature, Granada, Spain (2002) 700–709
11. Losee, R.: Learning syntactic rules and tags with genetic algorithms for information retrieval and filtering: An empirical basis for grammatical rules. Information Processing and Management **32** (1995) 185–197
12. De Pauw, G.: An Agent-Based Evolutionary Computing Approach to Memory-Based Syntactic Parsing of Natural Language. PhD thesis, University of Antwerp, Antwerp, Belgium (2002)
13. Marcus, M.P., Santorini, B., Marcinkiewicz, M.: Building a large annotated corpus of english: the penn treebank. Computational linguistics **19** (1993) 313–330 Reprinted in Susan Armstrong, ed. 1994, *Using large corpora*, Cambridge, MA: MIT Press, 273–290.
14. Bod, R.: Beyond Grammar—An Experience-Based Theory of Language. Cambridge University Press, Cambridge, England (1998)
15. Collins, M.: Head-driven Statistical Models for Natural Language Parsing. PhD thesis, University of Pennsylvania, Pennsylvania, USA (1999)
16. De Pauw, G.: Aspects of pattern-matching in dop. In: Proceedings of the 18th International Conference on Computational Linguistics. (2000) 236–242
17. Steels, L.: The origins of syntax in visually grounded robotic agents. Artificial Intelligence **103** (1998) 133–156
18. Chappelier, J.C., Rajman, M.: A generalized cyk algorithm for parsing stochastic cfg. In: Proceedings of Tabulation in Parsing and Deduction (TAPD'98), Paris (FRANCE) (1998) 133–137
19. van Zaanen, M., Adriaans, P.: Alignment-Based Learning versus EMILE: A comparison. In: Proceedings of the Belgian-Dutch Conference on Artificial Intelligence (BNAIC); Amsterdam, the Netherlands. (2001) 315–322
20. Clark, A.: Unsupervised induction of stochastic context-free grammars using distributional clustering. In Daelemans, W., Zajac, R., eds.: Proceedings of CoNLL-2001, Toulouse, France (2001) 105–112
21. Yuret, D.: Discovery of Linguistic Relations Using Lexical Attraction. PhD thesis, MIT, Cambridge, MA (1998)